(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) Publication number : **0 492 970 B1**

(12) **EUROPEAN PATENT SPECIFICATION**

(45) Date of publication of patent specification : **02.08.95 Bulletin 95/31**

(51) Int. Cl.[6] : **G06F 9/34**, G06F 9/455

(21) Application number : **91311773.5**

(22) Date of filing : **18.12.91**

(54) Method and apparatus for extending computer architecture from thirty-two to sixty-four bits.

(30) Priority : **21.12.90 US 632017**

(43) Date of publication of application : **01.07.92 Bulletin 92/27**

(45) Publication of the grant of the patent : **02.08.95 Bulletin 95/31**

(84) Designated Contracting States : **DE FR GB IT SE**

(56) References cited :
EP-A- 0 148 478
EP-A- 0 230 351
ELECTRONIC DESIGN vol. 33, no. 25. October 1985, HASBROUCK HEIGHTS, NEW JERSEY pages 161-174, R. AGARWAL ET AL. '32-bit microprocessor is a fine match for today's languages and operating systems'

(73) Proprietor : **SUN MICROSYSTEMS, INC.**
2550 Garcia Avenue
Mountain View, CA 94043 (US)

(72) Inventor : **Powell, Michael**
2305 Emerson Street
Palo Alto, California 94301 (US)
Inventor : **Cmelik, Robert**
1024 Chula Vista Terrace
Sunnyvale, California 94086 (US)
Inventor : **Kong, Shing**
P.O. Box 391379
Montain View, California 94039 (US)
Inventor : **Ditzel, David**
4010 Page Mill Road
Los Altos Hills, California 94022 (US)
Inventor : **Kelly, Edmund**
5277 Rio Grande Drive
San Jose, California 95136 (US)

(74) Representative : **Wombwell, Francis**
Potts, Kerr & Co.
15, Hamilton Square
Birkenhead Merseyside L41 6BR (GB)

EP 0 492 970 B1

Description

BACKGROUND OF THE INVENTION

1 . Field of the Invention:

This invention relates to computer systems and, more particularly, to methods and apparatus for extending computer architecture from thirty-two to sixty-four bits.

2. History of the Prior Art:

As computers have improved in speed and ability, there has been a constant demand for more addressable memory. The size of the address directly controls the size of memory which may be addressed. Each additional bit in an address doubles the amount of addressable memory. Thus, a system using sixty-four bit addresses provides two to the thirty-second power more addresses than does a system using thirty-two bit addresses. Consequently, researchers are today attempting to develop computer systems based on sixty-four bit architectures. On the other hand, very powerful computers exist today which use thirty-two bit addresses. A very large amount of very effective software exists for such systems. Most companies have more invested in their computer software than they do in their hardware. These thirty-two bit systems could effectively handle larger and larger problems if there were an easy way to simply address more memory. Presuming that the only effective way to obtain more memory is to design systems using larger (sixty-four bit) addresses, it would be very foolish and economically disastrous to simply discard all of the effort presently invested in the design of thirty-two bit systems and their software. For this reason, it is an a priori requirement that any new computer system based on a new memory size must be able to use the old programs on the new architecture. A primary question in the design of such a system is, therefore, how to change memory address size and still be able to use the old programs on the new architecture.

Two different tacks have been taken in resolving this problem so that a new system can run both old and new programs. One way to do this is to essentially provide two different architectures within the same machine for handling programs based on the different memory sizes and give the machine the ability to select one or the other architecture. This allows a machine to implement two different instruction sets. This is generally referred to as mode selection. One problem with this solution is that the need for two independent architectures must be perpetuated with each new series of machines; the manufacturer must continue to build systems including both thirty-two and sixty-four bit architectures for so long as the thirty-two bit programs are to be used. Another problem with this solution is that old and new procedures cannot communicate easily since they operate on different portions of the system. Digital Equipment Company (DEC) attempted this type of solution in changing from its PDP11 series of computers to its VAX series of computers. DEC set up microcode for operating both the PDP11 and the VAX instruction sets in its new VAX computers and used a mode switch which allowed selection between the two instruction sets in the machine.

A second solution to the problem is to design the new system so that the same hardware is able to handle both old and new processes using the same circuitry. Such a system is known from EP-A-0 148 478. This is a much more desirable solution. However, the prior art attempts to reach this solution have not, in fact, accomplished their purpose of eliminating hardware directed specifically to the different architectures. For example, Intel has followed this path in designing and improving its 80X86 line of computers. In general, Intel has continued the memory-handling hardware mechanisms for programs based on the old architecture and has added additional memory-handling hardware for programs based on the new architecture. The commands available to the old programs are essentially a subset of the entire set of commands available in the new machine. Although this solution allows the use of both old and new software by the same processor, it has provided no easy way to utilize old and new processes together. This method has also required the inclusion of the additional memory-handling hardware with each more advanced step, often to the detriment of the more advanced architecture. For example, the advanced Intel processors are still unable to deal with the larger address space except through the use of memory mapping hardware provided in the older machines.

Thus, there have been at least two different solutions to the problem. Each solution has not resolved the problem in a manner in which the new system is simply able to run either old or new programs using the new address space apparently without any great amount of new hardware except that necessary to allow the larger addresses to be utilized and without any rewriting of old software to fit the new hardware.

One especially difficult problem associated with providing such a system is caused by the manner in which an older system maintains its register files and stores the contents of those registers to memory and restores those contents during certain operations. Since a new system using a memory address size twice as large as an old system must necessarily use different space than the old system to store this information, some way must be found to allow the system to handle information for programs of both sizes correctly.

SUMMARY OF THE INVENTION

It is, therefore, an object of the present invention to provide a method and apparatus for allowing a computer system to run existing programs having a thirty-two bit word size and new programs having a sixty-four bit word size.

It is another more specific object of the present invention to provide a method and apparatus for allowing a computer system to store from register files and restore to register files from memory both existing programs having a thirty-two bit word size and new programs having a sixty-four bit word size without any substantial increase in hardware and without requiring a modification of the existing thirty-two bit software.

It is yet another more specific object of the present invention to provide a method and apparatus for allowing a computer system to save and restore state in response to the existence of traps for both existing programs having a thirty-two bit word size and new programs having a sixty-four bit word size.

These and other objects of the present invention are realized in a method and a system as disclosed in claims 1 and 17, respectively.

These and other objects and features of the invention will be better understood by reference to the detailed description which follows taken together with the drawings in which like elements are referred to by like designations throughout the several views.

BRIEF DESCRIPTION OF THE DRAWINGS

Figure 1 illustrates register files utilized in the thirty-two bit SPARC architecture and in the sixty-four bit architecture designed to replace the thirty-two bit architecture.

Figure 2 is an illustration of a region of memory used for saving a register file in a thirty-two bit architecture.

Figure 3 is an illustration of a region of memory used for saving a register file in a sixty-four bit architecture.

Figure 4 is an illustration of the steps used in practicing the method of this invention.

NOTATION AND NOMENCLATURE

Some portions of the detailed descriptions which follow are presented in terms of algorithms and symbolic representations of operations on data bits within a computer memory. These algorithmic descriptions and representations are the means used by those skilled in the data processing arts to most effectively convey the substance of their work to others skilled in the art. An algorithm is here, and generally, conceived to be a self-consistent sequence of steps leading to a desired result. The steps are those requiring physical manipulations of physical quantities. Usually, though not necessarily, these quantities take the form of electrical or magnetic signals capable of being stored, transferred, combined, compared, and otherwise manipulated. It has proven convenient at times, principally for reasons of common usage, to refer to these signals as bits, values, elements, symbols, characters, terms, numbers, or the like. It should be borne in mind, however, that all of these and similar terms are to be associated with the appropriate physical quantities and are merely convenient labels applied to these quantities.

Further, the manipulations performed are often referred to in terms, such as adding or comparing, which are commonly associated with mental operations performed by a human operator. No such capability of a human operator is necessary or desirable in most cases in any of the operations described herein which form part of the present invention; the operations are machine operations. Useful machines for performing the operations of the present invention include general purpose digital computers or other similar devices. In all cases the distinction between the method operations in operating a computer and the method of computation itself should be borne in mind. The present invention relates to apparatus and to method steps for operating a computer in processing electrical or other (e.g. mechanical, chemical) physical signals to generate other desired physical signals.

DETAILED DESCRIPTION OF THE INVENTION

Although the problem of changing a system from one which handles thirty-two bit addresses to one that handles sixty-four bit addresses seems simple, just change the size of the registers, it is in reality much more complicated because of the hardware and the software already in existence. The reasons for this complication may be better understood by referring to Figure 1 which illustrates register files utilized in thirty-two bit SPARC architecture and in sixty-four bit architecture designed to replace the thirty-two bit architecture. Machines using the SPARC architecture are designed, manufactured, and sold by Sun Microsystems, Inc., Mountain View, California. On the left are shown sixteen individual registers each capable of storing thirty-two bits of binary information as one and zero states. The bit positions within the individual registers are indicated at the top of the thirty-two bit register file. It should be noted that in the SPARC format the seventh register from the bottom is by convention referred to as the stack pointer, for it contains the stack address to which a portion of the register file is to be stored in memory in case of an interrupt.

The sixty-four bit register file shown to the right is similarly arranged, and contains sixteen individual registers each, however, capable of holding sixty-four bits of binary information. The bit positions for the

registers are indicated at the top of the sixty-four bit register file. The words stored in the sixty-four bit registers conform in all important respects to the words used in thirty-two bit SPARC architecture.

Those skilled in the art will understand that, in general, the data contained in thirty-two bit registers can be placed and used in sixty-four bit registers without affecting the results. This occurs because most architectures utilize two's complement arithmetic which, in general, overflows to the left. And the thirty-two bits used for commands in a thirty-two bit architecture can certainly be placed in the register space provided by a sixty-four bit architecture.

However, in running software under most architectures, it is continually necessary to switch from one procedure to another or from one program to another. Whenever there is an insufficient number of registers in the processor to handle a new procedure, the contents of at least some of the registers must be saved to memory so that the space they occupy may be used by the new procedure. Generally, whenever a program is operating and the system must stop a particular procedure and transfer to another procedure, it is necessary for the state of the present procedure (the information held in the register file regarding and being used by that procedure) to be saved for use upon return to the procedure. Then the registers must be restored for the new procedure, the new procedure must be run, the state from the old procedure retrieved and placed in the registers, and the old procedure restarted. This is true of most architectures no matter what format they run in.

The SPARC architecture, unlike many architectures, has a register file which contains many register windows in its integer processor. Each register window includes sixteen registers, eight of which are called LOCAL registers and eight of which are called IN registers. At least two register windows exist in any implementation of the SPARC architecture. The IN registers of one register window function as the OUT registers of the immediately preceding window so that the register windows may be thought to form a large ring. By switching from one window to the adjacent window when a program changes from one procedure to the next, the processor may run a number of procedures without having to save to memory the registers of a procedure which has been left and later having to restore the registers of that procedure from memory. This speeds the operation of the processor.

However, even a SPARC processor has a limited number of register windows; and, when that limit it reached, the registers of the next window in the ring must be saved to memory. In order to accomplish this, the SPARC architecture uses a window pointer. When the window pointer indicates that the register file is out of register windows and a new window is required, an overflow trap is generated which signifies that the contents of the registers in the next register window must be saved to memory. When this procedure being saved is to be run again, its register contents must be restored from memory. In a similar manner, when a procedure which has been saved is restored to a register window, that register window probably contains a procedure which must be saved before the procedure being restored may be placed in the registers of the window. To accomplish this, the window pointer is used to generate an underflow trap.

Thus, it may be seen that in both conventional architectures and SPARC architecture, a time ultimately comes when register files must be first saved to memory and then restored to the register file.

In order to save the state of a procedure in response to an overflow or underflow trap, a thirty-two bit SPARC architecture machine takes thirty-two bits from each of the registers of its current register window in the integer processor register file and stores that information in thirty-two bit save space in memory (on the "stack"). The hardware and the programs are adapted to cause this to happen. The system uses an address stored in the stack pointer register and places the information stored in the registers of the register file at addresses in memory beginning at that stack pointer address. Thus, the hardware and software combine to produce a particular amount of thirty-two bit save space for storing register state on the occurrence of the trap. This is, in general, the way any system based on a thirty-two bit architecture operates.

Such a thirty-two bit save space for a SPARC machine is illustrated in Figure 2. As is shown, the save space includes sixteen thirty-two bit words starting at the stack pointer address. It will be noted that additional space exists above the register file storage area in the save space for other purposes. It will also be noted that the address held in the stack pointer register of the register file illustrated in Figure .1 is stored in the seventh word space in the save space in memory. When the procedure is ready to restart, the data in the save area is returned to the register file used with the processor so that it may be utilized.

In order to operate with thirty-two bit procedures; the architecture of any new system must adhere to the conventions used by that code. Consequently, any new system must be able to save thirty-two bit information from an old procedure held in a register file to a thirty-two bit save area just as did the old architecture. It must also be able to retrieve that information and replace it correctly in the register file.

It will be noted that the save space in memory provided for thirty-two bit procedures is not arranged to handle sixty-four bit words. Yet it is necessary that the system store the longer words somewhere. Various arrangements for placing the sixty-four bit registers used by the new architecture can be visualized. For example, it would be possible to halve the sixty-four bit words and store the halves including the high

bits above the low bits in adjacent word spaces in the save space starting at the stack pointer address. However, when the system looked for saved data from which to restore the registers, the data would be in different positions in the thirty-two and sixty-four bit formats. If the system retrieved data expecting it to be thirty-two bit data and it were sixty-four bit data (or vice versa), the system could not operate correctly.

Thus it seems clear that even were there an arrangement to provide sufficient space to store sixty-four bit words, the system would have to understand whether that information was sixty-four bit information or thirty-two bit information in order to function properly. And even if the information were to be appropriately stored, the system must know how to restore the information when required.

This poses a number of questions. First, how does the system know whether the information in a register file is thirty-two or sixty-four bit information? Second, even if the system knows information is in a particular format, what do the bits mean? For example, the upper bits held in a register may be meaningless if the procedure is a thirty-two bit procedure, or they may be necessary information if the procedure is a sixty-four bit procedure. If the machine restores sixty-four bits to the registers and uses meaningless bits in a thirty-two bit procedure, the procedure is in great difficulty.

This problem is difficult where the registers and the storage are fixed in hardware. However, where the registers and the save space are delineated by software, the problem is much greater. For example, the conventions used in defining the operation of a particular system (such as the SPARC architecture) define save regions for the programs or procedures operating on the particular system. Each procedure sets up a register stack in this manner. It sets up a stack pointer which points to a register save area. The register save area and the position of the stack pointer cannot vary or the old programs will not work. Since the old procedures define the arrangement, all of the old procedures adapted to run on thirty-two bit machines would have to be changed in order to function correctly and know what to do with the information. It is simply impossible to correctly fix (translate) all of the old programs. Consequently, the changes made in changing the system to include sixty-four bit architecture must be those which can be made to the new system software to allow it to run with both the old and the new programs without the necessity of varying those programs.

The present invention solves the problem of storing and restoring both thirty-two and sixty-four bit information without any substantial increase in hardware and without requiring a modification of the existing thirty-two bit software. The present invention utilizes an indication at the procedure level to signal whether the procedure is a thirty-two or a sixty-four bit procedure. In order to accomplish this, the stack pointer is used to store the indication; this is a convenient place for the indication because the system must look to the stack pointer to find the address of the save space. On a trap requiring a register file save, the indication is looked at to determine whether the state is to be stored in thirty-two bit save space or sixty-four bit save space. In this manner, the save space for old procedures may utilize the same area of memory as would have been used under the old system software. Additional space may then be provided to store the registers of a sixty-four bit procedure.

Figure 1 illustrates how this may be accomplished in a system using the SPARC architecture. As is illustrated in Figure 1, the sixty-four bit registers contain the values of the thirty-two bit registers in the low order bits. Since each thirty-two bit procedure used in the SPARC architecture includes in the stack pointer register a stack pointer which indicates among other things the address in memory at which registers for this procedure are to be saved, the stack pointer register will always include this stack address. In order to indicate that the words of the particular procedure held in the registers are thirty-two or sixty-four bit words, bit 63 (the most significant bit) of the stack pointer is used. If the procedure is a thirty-two bit procedure, a zero is placed in this bit position. If the procedure is a sixty-four bit procedure, a one is placed in this bit position. In the SPARC architecture, the trap handler procedure (or hardware) is involved when an overflow trap indicates that the registers of a procedure presently running are to be saved. Whether done in hardware or software, the trap handler can see this bit in the stack pointer and act on it to save the registers in accordance with the particular procedure type.

It is important that this bit be used rather than some other bit. The stack pointer contains the address at which the save area is to be found. Since this address may be manipulated mathematically in addressing, the use of the highest order bit allows the stack pointer address to be added to or subtracted from without affecting the indication of the type of procedure held in bit 63. Since bit 63 is the most significant bit, it would not be expected to change at the low levels of addition or subtraction to be expected in address manipulations on the stack pointer.

It will be realized that since the SPARC architecture and almost all other architectures use two's complement arithmetic, the low order bits of the results of arithmetic operations do not depend on the high order bits of the operand; this allows arithmetic to be accomplished on the addresses in the stack pointer whether those addresses are in thirty-two bit or sixty-four bit code.

The arrangement described allows the system to

know the format of the data used by the procedure when a save operation is needed. In response to this indication, the system may place the information in the registers for a thirty-two bit Procedure in the save space illustrated in Figure 2. However, as discussed above, this space is not organized for the register information of a sixty-four bit procedure. Consequently, sixty-four bit functions or procedures define a new save area. That save area is illustrated in Figure 3. The area commences at the same stack pointer address and preempts the same register space as does the thirty-two bit area. However, the only information stored in this space is the lower order thirty-two bits of the stack pointer address; the other register save positions are left empty. Then beginning at the first position above the thirty-two bit save area, a completely new sixty-four bit save area is defined. In the preferred embodiment of the invention which is used in a SPARC architecture, the data from each register is stored in two thirty-two bit words adjacent one another, the low bits being stored below the high bits. A total of thirty-two word spaces, each of thirty-two bits, is thus available to store the register file for a sixty-four bit procedure.

It should be noted that the stack pointer is stored in unmodified form again in this sixty-four bit save area. The reason for this will become apparent from the following description.

In order to indicate that a saved procedure is either a thirty-two or a sixty-four bit procedure upon a restore, the indication is moved to the low thirty-two bits so that only the thirty-two bit save area need be searched. The bit indicates to the machine whether only the lower thirty-two bits of the state in a thirty-two bit procedure are to be used when restored or that sixty-four bits of the state in a sixty-four bit procedure are to be used when restored.

As explained above, the indication placed in the stack pointer register o6 is sufficient to indicate to the system whether the procedure being run in the register file is a thirty-two bit or a sixty-four bit procedure. However, the state stored in the save area must also be restored in order that it may be reused. There must be some way for the system to determine when it looks at the save area whether this is a thirty-two bit or sixty-four bit procedure. Since one of the requirements of the new design is that the thirty-two bit save area format not be changed, looking at the information in bit 63 of the stack pointer for a thirty-two bit procedure will tell nothing since this bit does not exist in the save area for such a thirty-two bit procedure.

Thus, there must be some way to determine from the lower thirty-two bits of the stack pointer the type of procedure which is involved. Fortunately, the values for stack pointer addresses are aligned on four byte boundaries in the SPARC architecture and other architectures as well. Thus the stack pointer register holds an address which includes a zero as its lowest order bit. Use of this bit to record the format indication stored in the stack pointer register at the high order bit allows the indication to be transferred to the thirty-two bit save area when the state of the registers is saved in both thirty-two bit and sixty-four bit save areas of memory. The state of the bit in the 63 bit position is therefore moved into the lowest order bit when the state of the register file is saved. In this manner, only the stack pointer of the thirty-two bit save area need be searched.

An underflow trap is used to restore the save area of memory. The underflow trap need simply look at the lowest order bit of the stack pointer to determine whether a thirty-two or sixty-four procedure is stored in the save area This tells the system the particular type of procedure being used. In the preferred embodiment, a zero is used to indicate a thirty-two bit procedure. Since thirty-two bit procedures already in use are not to be changed and a zero is normally found in the lowest order bit, this leaves the thirty-two bit procedure save area unaltered. On the other hand, a one in the lowest order bit is used to indicate a sixty-four bit procedure. Because a copy of the lower thirty-two bits of the sixty-four bit stack pointer (actually only the lowest order bit is necessary) is stored in the usual position for a thirty-two bit stack pointer, the sixty-four bit procedure responds to the same underflow trap without any change to the procedure other than the response to restore from different save positions once a sixty-four bit procedure has been detected by the trap.

Since the information in the stack pointer must be used by the procedure, if the procedure is a sixty-four bit procedure, the system leaves the stack pointer in the upper area in its original condition with a one in the lowest bit position so that no time is lost when the register file is restored. As is also illustrated in Figure 3, the one remains in bit position 63 designating a sixty-four bit procedure even though the lowest order bit of the stack pointer in the thirty-two bit save area provides the same information while the register file is stored in the save area.

It should be noted that one problem which exists with the use of the same sixty-four bit registers for both thirty-two and sixty-four bit procedures is that a register file executing a thirty-two bit procedure may include a number of higher order bits which are not related to the procedure. In some cases these bits can adversely affect the outcome of the procedure being run. For example, if the register contains an address to be used in a load or a store operation for a thirty-two bit procedure, if the upper bits are used, they produce the wrong address. Consequently, these bits must be masked off in some manner. In the preferred embodiment of the present invention, a mask register is used to store information relating to the particular bits of the address which are to be used. When data is loaded or stored using an address held in the reg-

ister file, the mask register is used to force the state of the upper bits which would otherwise carry invalid information. The details of such a mask register are described in more detail in EP patent application (Publication No. 0 492 970) entitled MASK REGISTER FOR COMPUTER PROCESSOR, Powell et al, filed on even date herewith and assigned to the assignee of the present invention.

Figure 4 illustrates and reiterates the steps used in practicing the method of this invention.

Although the present invention has been described in terms of a preferred embodiment, it will be appreciated that various modifications and alterations might be made by those skilled in the art.

Claims

1. A method for running computer procedures of different word sizes on a processor having registers designed to run procedures of a largest one of the word sizes comprising the step of placing an indication designating the word size of the procedure in at least one of a number of the registers of the processor, said method being characterised by the steps of providing separate save areas in memory for the register files of each of the word sizes, but all of the save areas including at least one portion at the same address, that portion being a portion of the save area of the procedure of a smallest one of the word sizes, placing the indication held in at least one of a number of the registers of the processor designating the word size of the procedure in the portion of the save areas having the same address each time a save operation occurs, and reviewing the indication in the portion of the save areas having the same address each time a restore operation occurs to determine the size of word in the procedure being restored.

2. A method for running computer procedures of different word sizes on a processor having registers designed to run procedures of a largest one of the word sizes as claimed in Claim 1 wherein the step of placing an indication designating the word size of the procedure in at least one of a number of the registers of the processor comprises placing an indication in a register containing a stack pointer address.

3. A method for running computer procedures of different word sizes on a processor having registers designed to run procedures of a largest one of the word sizes as claimed in Claim 2 wherein the step of placing an indication designating the word size of the procedure in a register containing a stack pointer address comprises placing an indication in a most significant bit position.

4. A method for running computer procedures of different word sizes on a processor having registers designed to run procedures of a largest one of the word sizes as claimed in Claim 3 wherein the step of placing an indication designating the word size of the procedure in a register containing a stack pointer address comprises placing a binary one indication in a most significant bit position to indicate a procedure of a largest one of the word sizes, and placing a binary zero indication in a most significant bit position to indicate a procedure of a smallest one of the word sizes.

5. A method for running computer procedures of different word sizes on a processor having registers designed to run procedures of a largest one of the word sizes as claimed in Claim 2 wherein the step of providing save areas in memory for the register files of each of the word sizes, further comprises providing a stack pointer address portion of the save areas as the at least one portion at the same address.

6. A method for running computer procedures of different word sizes on a processor having registers designed to run procedures of a largest one of the word sizes as claimed in Claim 5 wherein the step of placing the indication held in at least one of a number of the registers of the processor designating the word size of the procedure in the portion of the save areas having the same address each time a save operation occurs further comprises placing the indication in an otherwise insignificant bit of the portion holding the stack pointer address.

7. A method for running computer procedures of different word sizes on a processor having registers designed to run procedures of a largest one of the word sizes as claimed in Claim 6 wherein the step of placing the indication in an otherwise insignificant bit of the portion holding the stack pointer address comprises placing the bit in a lower order bit position of the stack pointer address.

8. A method for running computer procedures of different word sizes on a processor having registers designed to run procedures of a largest one of the word sizes as claimed in Claim 7 further comprises the step of restoring the information to the registers from the save area designated by the lower order bit position indication.

9. A method for running computer procedures of different word sizes on a processor having registers designed to run procedures of a largest one of the

word sizes as claimed in Claim 1 wherein the step of providing save areas in memory for the register files of each of the word sizes, further comprises providing a stack pointer address portion of the save areas as the at least one portion at the same address.

10. A method for running computer procedures of different word sizes on a processor having registers designed to run procedures of a largest one of the word sizes as claimed in Claim 9 wherein the step of placing an indication designating the word size of the procedure in at least one of a number of the registers of the processor comprises placing an indication in a register containing the stack pointer address.

11. A method for running computer procedures of different word sizes on a processor having registers designed to run procedures of a largest one of the word sizes as claimed in Claim 10 wherein the step of placing an indication designating the word size of the procedure in a register containing the stack pointer address comprises placing an indication in a most significant bit position.

12. A method for running computer procedures of different word sizes on a processor having registers designed to run procedures of a largest one of the word sizes as claimed in Claim 11 wherein the step of placing an indication designating the word size of the procedure in a register containing the stack pointer address comprises placing a binary one indication in a most significant bit position to indicate a procedure of a largest one of the word sizes, and placing a binary zero indication in a most significant bit position to indicate a procedure of a smallest one of the word sizes.

13. A method for running computer procedures of different word sizes on a processor having registers designed to run procedures of a largest one of the word sizes as claimed in Claim 12 further comprises the step of restoring the information to the registers from the save area designated by the indication.

14. A method for running computer procedures of different word sizes on a processor having registers designed to run procedures of a largest one of the word sizes as claimed in Claim 9 wherein the step of placing the indication held in at least one of a number of the registers of the processor designating the word size of the procedure in the portion of the save areas having the same address each time a save operation occurs further comprises placing the indication in an insignificant bit of the portion holding the stack pointer address.

15. A method for running computer procedures of different word sizes on a processor having registers designed to run procedures of a largest one of the word sizes as claimed in Claim 14 wherein the step of placing the indication in an insignificant bit of the portion holding the stack pointer address comprises placing the bit in a lowest order bit position of the stack pointer address.

16. A method for running computer procedures of different word sizes on a processor having registers designed to run procedures of a largest one of the word sizes as claimed in Claim 1 further comprises the step of restoring the information to the registers from the save area designated by the indication.

17. Apparatus for running computer procedures of different word sizes on a processor having registers (00...L7) designed to run procedures of a largest one of the word sizes comprising means for placing an indication designating the word size of the procedure in at least one of a number of the registers (06) of the processor, and characterised by means for providing separate save areas in memory for the register files of each of the word sizes, but all of the save areas including at least one portion at the same address, that portion being a portion of the save area of the procedure of a smallest one of the word sizes, means for placing the indication held in at least one of a number of the registers of the processor designating the word size of the procedure in the portion of the save areas having the same address each time a save operation occurs, and means for reviewing the indication in the portion of the save areas having the same address each time a restore operation occurs to determine the size of word in the procedure being restored.

18. Apparatus for running computer procedures of different word sizes on a processor having registers designed to run procedures of a largest one of the word sizes as claimed in Claim 17 wherein the means for placing an indication designating the word size of the procedure in at least one of a number of the registers of the processor comprises means for placing an indication in a register containing a stack pointer address.

19. Apparatus for running computer procedures of different word sizes on a processor having registers designed to run procedures of a largest one of the word sizes as claimed in Claim 18 wherein the means for placing an indication designating the word size of the procedure in a register containing a stack pointer address comprises means for placing an indication in a most significant bit

position.

20. Apparatus for running computer procedures of different word sizes on a processor having registers designed to run procedures of a largest one of the word sizes as claimed in Claim 18 wherein the means for placing an indication designating the word size of the procedure in a register containing a stack pointer address comprises means for placing a binary one indication in a most significant bit position to indicate a procedure of a largest one of the word sizes, and means for placing a binary zero indication in a most significant bit position to indicate a procedure of a smallest one of the word sizes.

21. Apparatus for running computer procedures of different word sizes on a processor having registers designed to run procedures of a largest one of the word sizes as claimed in Claim 18 wherein the means for providing save areas in memory for the register files of each of the word sizes, further comprises means for providing a stack pointer address portion of the save areas as the at least one portion at the same address.

22. Apparatus for running computer procedures of different word sizes on a processor having registers designed to run procedures of a largest one of the word sizes as claimed in Claim 21 wherein the means for placing the indication held in at least one of a number of the registers of the processor designating the word size of the procedure in the portion of the save areas having the same address each time a save operation occurs further comprises means for placing the indication in an otherwise insignificant bit of the portion holding the stack pointer address.

23. Apparatus for running computer procedures of different word sizes on a processor having registers designed to run procedures of a largest one of the word sizes as claimed in Claim 22 wherein the means for placing the indication in an otherwise insignificant bit of the portion holding the stack pointer address comprises means for placing the bit in a lower order bit position of the stack pointer address.

24. Apparatus for running computer procedures of different word sizes on a processor having registers designed to run procedures of a largest one of the word sizes as claimed in Claim 23 further comprises means for restoring the information to the registers from the save area designated by the lower order bit position indication.

25. Apparatus for running computer procedures of different word sizes on a processor having registers designed to run procedures of a largest one of the word sizes as claimed in Claim 17 wherein the means for providing save areas in memory for the register files of each of the word sizes, further comprises means for providing a stack pointer address portion of the save areas as the at least one portion at the same address

26. Apparatus for running computer procedures of different word sizes on a processor having registers designed to run procedures of a largest one of the word sizes as claimed in Claim 25 wherein the means for placing an indication designating the word size of the procedure in at least one of a number of the registers of the processor comprises means for placing an indication in a register containing the stack pointer address.

27. Apparatus for running computer procedures of different word sizes on a processor having registers designed to run procedures of a largest one of the word sizes as claimed in Claim 26 wherein the means for placing an indication designating the word size of the procedure in a register containing the stack pointer address comprises means for placing an indication in a most significant bit position.

28. Apparatus for running computer procedures of different word sizes on a processor having registers designed to run procedures of a largest one of the word sizes as claimed in Claim 27 wherein the means for placing an indication designating the word size of the procedure in a register containing the stack pointer address comprises means for placing a binary one indication in a most significant bit position to indicate a procedure of a largest one of the word sizes, and means for placing a binary zero indication in a most significant bit position to indicate a procedure of a smallest one of the word sizes.

29. Apparatus for running computer procedures of different word sizes on a processor having registers designed to run procedures of a largest one of the word sizes as claimed in Claim 28 further comprises means for restoring the information to the registers from the save area designated by the indication.

30. Apparatus for running computer procedures of different word sizes on a processor having registers designed to run procedures of a largest one of the word sizes as claimed in Claim 25 wherein the means for placing the indication held in at least one of a number of the registers of the processor designating the word size of the procedure

in the portion of the save areas having the same address each time a save operation occurs further comprises means for placing the indication in an otherwise insignificant bit of the portion holding the stack pointer address.

31. Apparatus for running computer procedures of different word sizes on a processor having registers designed to run procedures of a largest one of the word sizes as claimed in Claim 30 wherein the means for placing the indication in an otherwise insignificant bit of the portion holding the stack pointer address comprises means for placing the bit in a lower order bit position of the stack pointer address.

32. Apparatus for running computer procedures of different word sizes on a processor having registers designed to run procedures of a largest one of the word sizes as claimed in Claim 17 further comprises means for restoring the information to the registers from the save area designated by the indication.

Patentansprüche

1. Verfahren zum Abarbeiten von Computerprozeduren verschiedener Wortlängen auf einem Prozessor mit Registern, die so gestaltet sind, daß Prozeduren von einer größten Wortlänge abgearbeitet werden können, wobei das Verfahren den Schritt aufweist, daß ein die Wortlänge der Prozedur kennzeichnendes Kennzeichen in wenigstens einem von einer Mehrzahl der Register des Prozessors angeordnet wird, wobei das Verfahren durch die Schritte gekennzeichnet ist,

   daß separate Sicherungsbereiche im Speicher für die Registerdateien jeder der Wortlängen zur Verfügung gestellt werden, wobei jedoch alle Sicherungsbereiche wenigstens einen Abschnitt an der gleichen Adresse enthalten, wobei der Abschnitt ein Abschnitt des Sicherungsbereichs der Prozedur einer kleinsten Wortlänge ist,

   daß das in wenigstens einem der Mehrzahl der Register des Prozessors gehaltene, die Wortlänge der Prozedur kennzeichnende Kennzeichen in dem die gleiche Adresse aufweisenden Abschnitt des Sicherungsbereichs jedesmal dann angeordnet wird, wenn eine Sicherungsoperation auftritt, und

   daß das Kennzeichen in dem die gleiche Adresse aufweisenden Abschnitt des Sicherungsbereiches jedesmal dann überprüft wird, wenn eine Wiederherstellungsoperation auftritt, um die Wortlänge der wiederhergestellten Prozedur zu bestimmen.

2. Verfahren zum Abarbeiten von Computerprozeduren verschiedener Wortlängen auf einem Prozessor mit Registern, die zum Abarbeiten von Prozeduren einer größten Wortlänge konzipiert sind, nach Anspruch 1, wobei der Schritt des Anordnens eines die Wortlänge der Prozedur kennzeichnenden Kennzeichens in wenigstens einem von einer Mehrzahl der Register des Prozessors aufweist, daß ein Kennzeichen in einem eine Stapelspeicherzeiger-Adresse enthaltenden Register angeordnet wird.

3. Verfahren zum Abarbeiten von Computerprozeduren verschiedener Wortlängen auf einem Prozessor mit Registern, die zum Abarbeiten von Prozeduren einer größten Wortlänge konzipiert sind, nach Anspruch 2, wobei der Schritt des Anordnens eines die Wortlänge der Prozedur kennzeichnenden Kennzeichens in einem eine Stapelspeicherzeiger-Adresse enthaltenden Register aufweist, daß ein Kennzeichen in einer am höchsten bewerteten Bitposition angeordnet wird.

4. Verfahren zum Abarbeiten von Computerprozeduren verschiedener Wortlängen auf einem Prozessor mit Registern, die zum Abarbeiten von Prozeduren einer größten Wortlänge konzipiert sind, nach Anspruch 3, wobei der Schritt des Anordnens eines die Wortlänge der Prozedur kennzeichnenden Kennzeichens in einem eine Stapelspeicherzeiger-Adresse enthaltenden Register aufweist, daß eine binäre Eins in einer am höchsten bewerteten Bitposition angeordnet wird, um eine Prozedur einer größten Wortlänge anzuzeigen, und daß eine binäre Null in einer am höchsten bewerteten Position angeordnet wird, um eine Prozedur mit einer kleinsten Wortlänge anzuzeigen.

5. Verfahren zum Abarbeiten von Computerprozeduren verschiedener Wortlängen auf einem Prozessor mit Registern, die zum Abarbeiten von Prozeduren einer größten Wortlänge konzipiert sind, nach Anspruch 2, wobei der Schritt des Zur-Verfügung-Stellens von Sicherungsbereichen in dem Speicher für die Registerdateien jeder der Wortlängen außerdem aufweist, daß ein Stapelspeicherzeiger-Adreßabschnitt des Sicherungsbereichs als der wenigstens eine die gleiche Adresse aufweisende Abschnitt zur Verfügung gestellt wird.

6. Verfahren zum Abarbeiten von Computerprozeduren verschiedener Wortlängen auf einem Prozessor mit Registern, die zum Abarbeiten von Prozeduren mit einer größten Wortlänge konzipiert sind, nach Anspruch 5, wobei der Schritt des Anordnens des in wenigstens einem von einer

Mehrzahl von Registern des Prozessors gehaltenen, die Wortlänge der Prozedur kennzeichnenden Kennzeichens in dem die gleiche Adresse aufweisenden Abschnitt des Sicherungsbereichs jedesmal, wenn eine Sicherungsoperation auftritt, außerdem aufweist, daß das Kennzeichen in einem ansonsten bedeutungslosen Bit des Abschnitts angeordnet wird, der die Stapelspeicherzeiger-Adresse enthält.

7. Verfahren zum Abarbeiten von Computerprozeduren verschiedener Wortlängen auf einem Prozessor mit Registern, die zum Abarbeiten von Prozeduren einer größten Wortlänge konzipiert sind, nach Anspruch 6, wobei der Schritt des Anordnens des Kennzeichens in einem ansonsten bedeutungslosen Bit des die Stapelspeicherzeiger-Adresse enthaltenden Abschnitts aufweist, daß das Bit in einer Bitposition niedriger Ordnung der Stapelspeicherzeiger-Adresse angeordnet wird.

8. Verfahren zum Abarbeiten von Computerprozeduren verschiedener Wortlängen auf einem Prozessor mit Registern, die zum Abarbeiten von Prozeduren einer größten Wortlänge konzipiert sind, nach Anspruch 7, das außerdem den Schritt aufweist, daß die Informationen in den Registern aus dem von dem Kennzeichen an der Bitposition niedriger Ordnung gekennzeichneten Sicherungsbereich wiederhergestellt werden.

9. Verfahren zum Abarbeiten von Computerprozeduren verschiedener Wortlängen auf einem Prozessor mit Registern, die zum Abarbeiten von Prozeduren einer größten Wortlänge konzipiert sind, nach Anspruch 1, wobei der Schritt des Zur-Verfügung-Stellens von Sicherungsbereichen in dem Speicher für die Registerdateien jeder der Wortlängen außerdem aufweist, daß ein Stapelspeicherzeiger-Adreßabschnitt des Sicherungsbereichs als der wenigstens eine die gleiche Adresse aufweisende Abschnitt zur Verfügung gestellt wird.

10. Verfahren zum Abarbeiten von Computerprozeduren verschiedener Wortlängen auf einem Prozessor mit Registern, die zum Abarbeiten von Prozeduren einer größten Wortlänge konzipiert sind, nach Anspruch 9, wobei der Schritt des Anordnens eines die Wortlänge der Prozedur kennzeichnenden Kennzeichens in wenigstens einem einer Anzahl von Registern des Prozessors aufweist, daß ein Kennzeichen in einem Register angeordnet wird, das die Stapelspeicherzeiger-Adresse enthält.

11. Verfahren zum Abarbeiten von Computerprozeduren verschiedener Wortlängen auf einem Prozessor mit Registern, die zum Abarbeiten von Prozeduren einer größten Wortlänge konzipiert sind, nach Anspruch 10, wobei der Schritt des Anordnens eines die Wortlänge der Prozedur kennzeichnenden Kennzeichens in einem Register, das die Stapelspeicherzeiger-Adresse enthält, aufweist, daß ein Kennzeichen in einer am höchsten bewerteten Bitposition angeordnet wird.

12. Verfahren zum Abarbeiten von Computerprozeduren verschiedener Wortlängen auf einem Prozessor mit Registern, die zum Abarbeiten von Prozeduren einer größten Wortlänge konzipiert sind, nach Anspruch 11, wobei der Schritt des Anordnens eines die Wortlänge der Prozedur kennzeichnenden Kennzeichens in einem Register, das die Stapelspeicherzeiger-Adresse enthält, aufweist, daß eine binäre Eins in einer am höchsten bewerteten Bitposition angeordnet wird, um eine Prozedur einer größten Wortlänge anzuzeigen, und daß eine binäre Null in einer am höchsten bewerteten Bitposition angeordnet wird, um eine Prozedur einer kleinsten Wortlänge anzuzeigen.

13. Verfahren zum Abarbeiten von Computerprozeduren verschiedener Wortlängen auf einem Prozessor mit Registern, die zum Abarbeiten von Prozeduren einer größten Wortlänge konzipiert sind, nach Anspruch 12, wobei das Verfahren außerdem den Schritt aufweist, daß die Informationen in den Registern aus dem von dem Kennzeichen gekennzeichneten Sicherungsbereich wiederhergestellt werden.

14. Verfahren zum Abarbeiten von Computerprozeduren verschiedener Wortlängen auf einem Prozessor mit Registern, die zum Abarbeiten von Prozeduren einer größten Wortlänge konzipiert sind, nach Anspruch 9, wobei der Schritt des Anordnens des in wenigstens einem von einer Anzahl von Registern des Prozessors gehaltenen, die Wortlänge der Prozedur kennzeichnenden Kennzeichens in dem die gleiche Adresse aufweisenden Abschnitt des Sicherungsbereichs jedesmal, wenn eine Sicherungsoperation auftritt, außerdem aufweist, daß das Kennzeichen in einem bedeutungslosen Bit des Abschnitts angeordnet wird, der die Stapelspeicherzeiger-Adresse hält.

15. Verfahren zum Abarbeiten von Computerprozeduren verschiedener Wortlängen auf einem Prozessor mit Registern, die zum Abarbeiten von Prozeduren einer größten Wortlänge konzipiert sind, nach Anspruch 14, wobei der Schritt des Anordnens des Kennzeichens in einem bedeu-

tungslosen Bit des Abschnitts, der die Stapelspeicherzeiger-Adresse hält, aufweist, daß das Bit in einer Bitposition niedrigster Ordnung der Stapelspeicherzeiger-Adresse angeordnet wird.

16. Verfahren zum Abarbeiten von Computerprozeduren verschiedener Wortlängen auf einem Prozessor mit Registern, die zum Abarbeiten von Prozeduren einer größten Wortlänge konzipiert sind, nach Anspruch 1, wobei das Verfahren außerdem den Schritt aufweist, daß die Informationen in den Registern aus dem von dem Kennzeichen gekennzeichneten Sicherungsbereich wiederhergestellt werden.

17. Einrichtung zum Abarbeiten von Computerprozeduren verschiedener Wortlängen auf einem Prozessor mit Registern (00...L7), die zum Abarbeiten von Prozeduren einer größten Wortlänge konzipiert sind, aufweisend

Mittel zum Anordnen eines die Wortlänge der Prozedur kennzeichnenden Kennzeichens in wenigstens einem von einer Anzahl von Registern (06) des Prozessors, und gekennzeichnet durch

Mittel zum Zur-Verfügung-Stellen separater Sicherungsbereiche in dem Speicher für die Registerdateien jeder der Wortlängen, wobei jedoch alle Sicherungsbereiche wenigstens einen Abschnitt an der gleichen Adresse enthalten, wobei der Abschnitt ein Abschnitt des Sicherungsbereichs der Prozedur einer kleinsten Wortlänge ist,

Mittel zum Anordnen des in wenigstens einem einer Anzahl der Register des Prozessors gehaltenen, die Wortlänge der Prozedur kennzeichnenden Kennzeichens in dem die gleiche Adresse aufweisenden Abschnitt des Sicherungsbereichs jedesmal, wenn eine Sicherungsoperation auftritt, und

Mittel zum Überprüfen des Kennzeichens in dem die gleiche Adresse aufweisenden Abschnitt des Sicherungsbereichs jedesmal, wenn eine Wiederherstellungsoperation auftritt, um die Wortlänge in der wiederhergestellten Prozedur zu bestimmen.

18. Einrichtung zum Abarbeiten von Computerprozeduren verschiedener Wortlängen auf einem Prozessor mit Registern, die zum Abarbeiten von Prozeduren einer größten Wortlänge konzipiert sind, nach Anspruch 17, wobei die Mittel zum Anordnen eines die Wortlänge der Prozedur kennzeichnenden Kennzeichens in wenigstens einem von einer Anzahl von Registern des Prozessors Mittel zum Anordnen eines Kennzeichens in einem eine Stapelspeicherzeiger-Adresse enthaltenden Register aufweisen.

19. Einrichtung zum Abarbeiten von Computerprozeduren verschiedener Wortlängen auf einem Prozessor mit Registern, die zum Abarbeiten von Prozeduren einer größten Wortlänge konzipiert sind, nach Anspruch 18, wobei die Mittel zum Anordnen eines die Wortlänge der Prozedur kennzeichnenden Kennzeichens in einem eine Stapelspeicherzeiger-Adresse enthaltenden Register Mittel zum Anordnen eines Kennzeichens in einer am höchsten bewerteten Bitposition aufweisen.

20. Einrichtung zum Abarbeiten von Computerprozeduren verschiedener Wortlängen auf einem Prozessor mit Registern, die zum Abarbeiten von Prozeduren einer größten Wortlänge konzipiert sind, nach Anspruch 18, wobei die Mittel zum Anordnen eines die Wortlänge der Prozedur kennzeichnenden Kennzeichens in einem eine Stapelspeicherzeiger-Adresse enthaltenden Register Mittel zum Anordnen einer binären Eins in einer am höchsten bewerteten Bitposition zum Anzeigen einer Prozedur einer größten Wortlänge und Mittel zum Anordnen einer binären Null in eine am höchsten bewertete Bitpostion zum Anzeigen einer Prozedur einer kleinsten Wortlänge aufweisen.

21. Einrichtung zum Abarbeiten von Computerprozeduren verschiedener Wortlängen auf einem Prozessor mit Registern, die zum Abarbeiten von Prozeduren einer größten Wortlänge konzipiert sind, nach Anspruch 18, wobei die Mittel zum Zur-Verfügung-Stellen der Sicherungsbereiche in dem Speicher für Registerdateien jeder der Wortlängen außerdem Mittel zum Zur-Verfügung-Stellen eines Stapelspeicherzeiger-Adreßabschnitts des Sicherungsbereichs als der wenigstens eine die gleiche Adresse aufweisende Abschnitt aufweisen.

22. Einrichtung zum Abarbeiten von Computerprozeduren verschiedener Wortlängen auf einem Prozessor mit Registern, die zum Abarbeiten von Prozeduren einer größten Wortlänge konzipiert sind, nach Anspruch 21, wobei die Mittel zum Anordnen der in wenigstens einem einer Anzahl von Registern des Prozessors gehaltenen, die Wortlänge der Prozedur kennzeichnenden Kennzeichens in dem die gleiche Adresse aufweisenden Abschnitt des Sicherungsbereiches jedesmal, wenn eine Sicherungsoperation auftritt, außerdem Mittel zum Anordnen des Kennzeichens in einem ansonsten bedeutungslosen Bit des Abschnitts, der die Stapelspeicherzeiger-Adresse hält, aufweisen.

23. Einrichtung zum Abarbeiten von Computerprozeduren verschiedener Wortlängen auf einem Pro-

zessor mit Registern, die zum Abarbeiten von Prozeduren einer größten Wortlänge konzipiert sind, nach Anspruch 22, wobei die Mittel zum Anordnen des Kennzeichens in einem ansonsten bedeutungslosen Bit des Abschnitts, der die Stapelspeicherzeiger-Adresse enthält, Mittel zum Anordnen des Bits in einer Bitposition niedriger Ordnung der Stapelspeicherzeiger-Adresse aufweisen.

24. Einrichtung zum Abarbeiten von Computerprozeduren verschiedener Wortlängen auf einem Prozessor mit Registern, die zum Abarbeiten von Prozeduren einer größten Wortlänge konzipiert sind, nach Anspruch 23, wobei die Einrichtung außerdem Mittel zum Wiederherstellen von Informationen in den Registern aus dem von dem Kennzeichen an der Bitposition niedriger Ordnung gekennzeichneten Sicherungsbereich aufweisen.

25. Einrichtung zum Abarbeiten von Computerprozeduren verschiedener Wortlängen auf einem Prozessor mit Registern, die zum Abarbeiten von Prozeduren einer größten Wortlänge konzipiert sind, nach Anspruch 17, wobei die Mittel zum Zur-Verfügung-Stellen von Sicherungsbereichen in dem Speicher für die Registerdateien jeder der Wortlängen außerdem Mittel aufweisen, um einen Stapelspeicherzeiger-Adreßabschnitt der Sicherungsbereiche als den wenigstens einen die gleiche Adresse aufweisenden Abschnitt zur Verfügung zu stellen.

26. Einrichtung zum Abarbeiten von Computerprozeduren verschiedener Wortlängen auf einem Prozessor mit Registern, die zum Abarbeiten von Prozeduren einer größten Wortlänge konzipiert sind, nach Anspruch 25, wobei die Mittel zum Anordnen eines die Wortlänge der Prozedur kennzeichnenden Kennzeichens in wenigstens einem von einer Anzahl von Registern des Prozessors Mittel zum Anordnen eines Kennzeichens in einem die Stapelspeicherzeiger-Adresse enthaltenden Register aufweisen.

27. Einrichtung zum Abarbeiten von Computerprozeduren verschiedener Wortlängen auf einem Prozessor mit Registern, die zum Abarbeiten von Prozeduren einer größten Wortlänge konzipiert sind, nach Anspruch 26, wobei die Mittel zum Anordnen eines die Wortlänge der Prozedur kennzeichnenden Kennzeichens in einem Register, das die Stapelspeicherzeiger-Adresse enthält, Mittel zum Anordnen eines Kennzeichens in einer am höchsten bewerteten Bitposition aufweisen.

28. Einrichtung zum Abarbeiten von Computerprozeduren verschiedener Wortlängen auf einem Prozessor mit Registern, die zum Abarbeiten von Prozeduren einer größten Wortlänge konzipiert sind, nach Anspruch 27, wobei die Mittel zum Anordnen eines die Wortlänge der Prozedur kennzeichnenden Kennzeichens in einem die Stapelspeicherzeiger-Adresse enthaltenden Register Mittel zum Anordnen einer binären Eins in einer am höchsten bewerteten Bitposition zum Anzeigen einer Prozedur einer größten Wortlänge und Mittel zum Anordnen einer binären Null in einer am höchsten bewerteten Bitposition zum Anzeigen einer Prozedur einer kleinsten Wortlänge aufweisen.

29. Einrichtung zum Abarbeiten von Computerprozeduren verschiedener Wortlängen auf einem Prozessor mit Registern, die zum Abarbeiten von Prozeduren einer größten Wortlänge konzipiert sind, nach Anspruch 28, wobei die Einrichtung außerdem Mittel zum Wiederherstellen von Informationen in den Registern aus dem von dem Kennzeichen gekennzeichneten Sicherungsbereich aufweist.

30. Einrichtung zum Abarbeiten von Computerprozeduren verschiedener Wortlängen auf einem Prozessor mit Registern, die zum Abarbeiten von Prozeduren einer größten Wortlänge konzipiert sind, nach Anspruch 25, wobei die Mittel zum Anordnen des in wenigstens einem einer Anzahl von Registern des Prozessors gehaltenen, die Wortlänge der Prozedur kennzeichnenden Kennzeichens in dem die gleiche Adresse aufweisenden Abschnitt der Sicherungsbereiche jedesmal, wenn eine Sicherungsoperation auftritt, außerdem Mittel zum Anordnen des Kennzeichens in einem ansonsten bedeutungslosen Bit des Abschnittes, der die Stapelspeicherzeiger-Adresse hält, aufweisen.

31. Einrichtung zum Abarbeiten von Computerprozeduren verschiedener Wortlängen auf einem Prozessor mit Registern, die zum Abarbeiten von Prozeduren einer größten Wortlänge konzipiert sind, nach Anspruch 30, wobei die Mittel zum Anordnen des Kennzeichens in einem ansonsten bedeutungslosen Bit des Abschnitts, der die Stapelspeicherzeiger-Adresse hält, Mittel zum Anordnen des Bits in einer Bitposition niedriger Ordnung der Stapelspeicherzeiger-Adresse aufweisen.

32. Einrichtung zum Abarbeiten von Computerprozeduren verschiedener Wortlängen auf einem Prozessor mit Registern, die zum Abarbeiten von Prozeduren einer größten Wortlänge konzipiert sind, nach Anspruch 17, wobei die Einrichtung

außerdem Mittel zum Wiederherstellen von Informationen in den Registern aus dem von dem Kennzeichen gekennzeichneten Sicherungsbereich aufweist.

Revendications

1. Procédé pour exécuter des procédures informatiques de tailles de mot différentes dans un processeur ayant des registres conçus pour exécuter des procédures avec la plus grande des tailles de mot comprenant l'étape consistant à placer une indication désignant la taille de mot de la procédure dans au moins un registre d'un certain nombre de registres du processeur, ledit procédé étant caractérisé par les étapes consistant à fournir des zones de sauvegarde séparées dans une mémoire pour les fichiers de registres de chacune des tailles de mot, toutes les zones de sauvegarde incluant toutefois au moins une portion à la même adresse, cette portion étant une portion de la zone de sauvegarde de la procédure avec la plus petite des tailles de mot, à placer l'indication conservée dans au moins un registre d'un certain nombre de registres du processeur désignant la taille de mot de la procédure dans la portion des zones de sauvegarde ayant la même adresse à chaque fois qu'une opération de sauvegarde intervient, et à relire l'indication dans la portion des zones de sauvegarde ayant la même adresse à chaque fois qu'une opération de restauration intervient afin de déterminer la taille de mot dans la procédure en cours de restauration.

2. Procédé pour exécuter des procédures informatiques de tailles de mot différentes dans un processeur ayant des registres conçus pour exécuter des procédures avec la plus grande des tailles de mot selon la revendication 1, dans lequel l'étape consistant à placer une indication désignant la taille de mot de la procédure dans au moins un registre d'un certain nombre de registres du processeur comprend la mise en place d'une indication dans un registre contenant une adresse de pointeur de pile.

3. Procédé pour exécuter des procédures informatiques de tailles de mot différentes dans un processeur ayant des registres conçus pour exécuter des procédures avec la plus grande des tailles de mot selon la revendication 2, dans lequel l'étape consistant à placer une indication désignant la taille de mot de la procédure dans un registre contenant une adresse de pointeur de pile comprend la mise en place d'une indication dans une position binaire la plus significative.

4. Procédé pour exécuter des procédures informatiques de tailles de mot différentes dans un processeur ayant des registres conçus pour exécuter des procédures avec la plus grande des tailles de mot selon la revendication 3, dans lequel l'étape consistant à placer une indication désignant la taille de mot de la procédure dans un registre contenant une adresse de pointeur de pile comprend la mise en place d'une indication binaire "1" dans une position binaire la plus significative afin d'indiquer une procédure avec la plus grande des tailles de mot, et la mise en place d'une indication binaire "0" dans une position binaire la plus significative afin d'indiquer une procédure avec la plus petite des tailles de mot.

5. Procédé pour exécuter des procédures informatiques de tailles de mot différentes dans un processeur ayant des registres conçus pour exécuter des procédures avec la plus grande des tailles de mot selon la revendication 2, dans lequel l'étape consistant à fournir des zones de sauvegarde dans une mémoire pour les fichiers de registres de chacune des tailles de mot, comprend également la fourniture d'une portion d'adresse de pointeur de pile des zones de sauvegarde comme étant la au moins une portion à la même adresse.

6. Procédé pour exécuter des procédures informatiques de tailles de mot différentes dans un processeur ayant des registres conçus pour exécuter des procédures avec la plus grande des tailles de mot selon la revendication 5, dans lequel l'étape consistant à placer l'indication conservée dans au moins un registre d'un certain nombre de registres du processeur désignant la taille de mot de la procédure dans la portion des zones de sauvegarde ayant la même adresse à chaque fois qu'une opération de sauvegarde intervient comprend également la mise en place d'une indication dans un bit autrement non-significatif de la portion conservant l'adresse de pointeur de pile.

7. Procédé pour exécuter des procédures informatiques de tailles de mot différentes dans un processeur ayant des registres conçus pour exécuter des procédures avec la plus grande des tailles de mot selon la revendication 6, dans lequel l'étape consistant à placer l'indication dans un bit autrement non-significatif de la portion conservant l'adresse de pointeur de pile comprend la mise en place du bit dans une position binaire de poids faible de l'adresse de pointeur de pile.

8. Procédé pour exécuter des procédures informatiques de tailles de mot différentes dans un pro-

cesseur ayant des registres conçus pour exécuter des procédures avec la plus grande des tailles de mot selon la revendication 7, comprenant également l'étape consistant à restaurer l'information destinée aux registres à partir de la zone de sauvegarde désignée par l'indication à la position binaire de poids faible.

9. Procédé pour exécuter des procédures informatiques de tailles de mot différentes dans un processeur ayant des registres conçus pour exécuter des procédures avec la plus grande des tailles de mot selon la revendication 1, dans lequel l'étape consistant à fournir des zones de sauvegarde dans une mémoire pour les fichiers de registres de chacune des tailles de mot comprend également la fourniture d'une portion d'adresse de pointeur de pile des zones de sauvegarde comme étant la au moins une portion à la même adresse.

10. Procédé pour exécuter des procédures informatiques de tailles de mot différentes dans un processeur ayant des registres conçus pour exécuter des procédures avec la plus grande des tailles de mot selon la revendication 9, dans lequel l'étape consistant à placer une indication désignant la taille de mot de la procédure dans au moins un registre d'un certain nombre de registres du processeur comprend la mise en place d'une indication dans un registre contenant l'adresse de pointeur de pile.

11. Procédé pour exécuter des procédures informatiques de tailles de mot différentes dans un processeur ayant des registres conçus pour exécuter des procédures avec la plus grande des tailles de mot selon la revendication 10, dans lequel l'étape consistant à placer une indication désignant la taille de mot de la procédure dans un registre contenant l'adresse de pointeur de pile comprend la mise en place d'une indication dans une position binaire la plus significative.

12. Procédé pour exécuter des procédures informatiques de tailles de mot différentes dans un processeur ayant des registres conçus pour exécuter des procédures avec la plus grande des tailles de mot selon la revendication 11, dans lequel l'étape consistant à placer une indication désignant la taille de mot de la procédure dans un registre contenant l'adresse de pointeur de pile comprend la mise en place d'une indication binaire "1" dans une position binaire la plus significative afin d'indiquer une procédure avec la plus grande des tailles de mot, et la mise en place d'une indication binaire "0" dans une position binaire la plus significative afin d'indiquer une procédure avec la plus petite des tailles de mot.

13. Procédé pour exécuter des procédures informatiques de tailles de mot différentes dans un processeur ayant des registres conçus pour exécuter des procédures avec la plus grande des tailles de mot selon la revendication 12, comprenant également l'étape consistant à restaurer l'information destinée aux registres à partir de la zone de sauvegarde désignée par l'indication.

14. Procédé pour exécuter des procédures informatiques de tailles de mot différentes dans un processeur ayant des registres conçus pour exécuter des procédures avec la plus grande des tailles de mot selon la revendication 9, dans lequel l'étape consistant à placer l'indication conservée dans au moins un registre d'un certain nombre de registres du processeur désignant la taille de mot de la procédure dans la portion des zones de sauvegarde ayant la même adresse à chaque fois qu'une opération de sauvegarde intervient comprend également la mise en place d'une indication dans un bit non-significatif de la portion conservant l'adresse de pointeur de pile.

15. Procédé pour exécuter des procédures informatiques de tailles de mot différentes dans un processeur ayant des registres conçus pour exécuter des procédures avec la plus grande des tailles de mot selon la revendication 14, dans lequel l'étape consistant à placer l'indication dans un bit non-significatif de la portion conservant l'adresse de pointeur de pile comprend la mise en place du bit dans une position binaire de plus faible poids de l'adresse de pointeur de pile.

16. Procédé pour exécuter des procédures informatiques de tailles de mot différentes dans un processeur ayant des registres conçus pour exécuter des procédures avec la plus grande des tailles de mot selon la revendication 1, comprenant également l'étape consistant à restaurer l'information destinée aux registres à partir de la zone de sauvegarde désignée par l'indication.

17. Dispositif pour exécuter des procédures informatiques de tailles de mot différentes dans un processeur ayant des registres (00...L7) conçus pour exécuter des procédures avec la plus grande des tailles de mot comprenant des moyens pour placer une indication désignant la taille de mot de la procédure dans au moins un registre d'un certain nombre de registres (06) du processeur, caractérisé par des moyens pour fournir des zones de sauvegarde séparées dans une mémoire pour les fichiers de registres de chacune des tailles de mot, toutes les zones de sauvegarde in-

cluant toutefois au moins une portion à la même adresse, cette portion étant une portion de la zone de sauvegarde de la procédure avec la plus petite des tailles de mot, des moyens pour placer l'indication conservée dans au moins un registre d'un certain nombre de registres du processeur désignant la taille de mot de la procédure dans la portion des zones de sauvegarde ayant la même adresse à chaque fois qu'une opération de sauvegarde intervient, et des moyens pour relire l'indication dans la portion des zones de sauvegarde ayant la même adresse à chaque fois qu'une opération de restauration intervient afin de déterminer la taille de mot dans la procédure en cours de restauration.

18. Dispositif pour exécuter des procédures informatiques de tailles de mot différentes dans un processeur ayant des registres conçus pour exécuter des procédures avec la plus grande des tailles de mot selon la revendication 17, dans lequel les moyens pour placer une indication désignant la taille de mot de la procédure dans au moins un registre d'un certain nombre de registres du processeur comprennent des moyens pour placer une indication dans un registre contenant une adresse de pointeur de pile.

19. Dispositif pour exécuter des procédures informatiques de tailles de mot différentes dans un processeur ayant des registres conçus pour exécuter des procédures avec la plus grande des tailles de mot selon la revendication 18, dans lequel les moyens pour placer une indication désignant la taille de mot de la procédure dans un registre contenant une adresse de pointeur de pile comprennent des moyens pour placer une indication dans une position binaire la plus significative.

20. Dispositif pour exécuter des procédures informatiques de tailles de mot différentes dans un processeur ayant des registres conçus pour exécuter des procédures avec la plus grande des tailles de mot selon la revendication 18, dans lequel les moyens pour placer une indication désignant la taille de mot de la procédure dans un registre contenant une adresse de pointeur de pile comprennent des moyens pour placer une indication binaire "1" dans une position binaire la plus significative afin d'indiquer une procédure avec la plus grande des tailles de mot, et des moyens pour placer une indication binaire "0" dans une position binaire la plus significative afin d'indiquer une procédure avec la plus petite des tailles de mot.

21. Dispositif pour exécuter des procédures informatiques de tailles de mot différentes dans un processeur ayant des registres conçus pour exécuter des procédures avec la plus grande des tailles de mot selon la revendication 18, dans lequel les moyens pour fournir des zones de sauvegarde dans une mémoire pour les fichiers de registres de chacune des tailles de mot comprennent également des moyens pour fournir une portion d'adresse de pointeur de pile des zones de sauvegarde comme étant la au moins une portion à la même adresse.

22. Dispositif pour exécuter des procédures informatiques de tailles de mot différentes dans un processeur ayant des registres conçus pour exécuter des procédures avec la plus grande des tailles de mot selon la revendication 21, dans lequel les moyens pour placer l'indication conservée dans au moins un registre d'un certain nombre de registres du processeur désignant la taille de mot de la procédure dans la portion des zones de sauvegarde ayant la même adresse à chaque fois qu'une opération de sauvegarde intervient comprennent également des moyens pour placer une indication dans un bit autrement non-significatif de la portion conservant l'adresse de pointeur de pile.

23. Dispositif pour exécuter des procédures informatiques de tailles de mot différentes dans un processeur ayant des registres conçus pour exécuter des procédures avec la plus grande des tailles de mot selon la revendication 22, dans lequel les moyens pour placer l'indication dans un bit autrement non-significatif de la portion conservant l'adresse de pointeur de pile comprennent des moyens pour placer le bit dans une position binaire de poids faible de l'adresse de pointeur de pile.

24. Dispositif pour exécuter des procédures informatiques de tailles de mot différentes dans un processeur ayant des registres conçus pour exécuter des procédures avec la plus grande des tailles de mot selon la revendication 23, comprenant également des moyens pour restaurer l'information destinée aux registres à partir de la zone de sauvegarde désignée par l'indication à la position binaire de poids faible.

25. Dispositif pour exécuter des procédures informatiques de tailles de mot différentes dans un processeur ayant des registres conçus pour exécuter des procédures avec la plus grande des tailles de mot selon la revendication 17, dans lequel les moyens pour fournir des zones de sauvegarde dans une mémoire pour les fichiers de registres de chacune des tailles de mot comprennent également des moyens pour fournir une portion

d'adresse de pointeur de pile des zones de sauvegarde comme étant la au moins une portion à la même adresse.

26. Dispositif pour exécuter des procédures informatiques de tailles de mot différentes dans un processeur ayant des registres conçus pour exécuter des procédures avec la plus grande des tailles de mot selon la revendication 25, dans lequel les moyens pour placer une indication désignant la taille de mot de la procédure dans au moins un registre d'un certain nombre de registres du processeur comprennent des moyens pour placer une indication dans un registre contenant l'adresse de pointeur de pile.

27. Dispositif pour exécuter des procédures informatiques de tailles de mot différentes dans un processeur ayant des registres conçus pour exécuter des procédures avec la plus grande des tailles de mot selon la revendication 26, dans lequel les moyens pour placer une indication désignant la taille de mot de la procédure dans un registre contenant l'adresse de pointeur de pile comprennent des moyens pour placer une indication dans une position binaire la plus significative.

28. Dispositif pour exécuter des procédures informatiques de tailles de mot différentes dans un processeur ayant des registres conçus pour exécuter des procédures avec la plus grande des tailles de mot selon la revendication 27, dans lequel les moyens pour placer une indication désignant la taille de mot de la procédure dans un registre contenant l'adresse de pointeur de pile comprennent des moyens pour placer une indication binaire "1" dans une position binaire la plus significative afin d'indiquer une procédure avec la plus grande des tailles de mot, et des moyens pour placer une indication binaire "0" dans une position binaire la plus significative afin d'indiquer une procédure avec la plus petite des tailles de mot.

29. Dispositif pour exécuter des procédures informatiques de tailles de mot différentes dans un processeur ayant des registres conçus pour exécuter des procédures avec la plus grande des tailles de mot selon la revendication 28, comprenant également des moyens pour restaurer l'information destinée aux registres à partir de la zone de sauvegarde désignée par l'indication.

30. Dispositif pour exécuter des procédures informatiques de tailles de mot différentes dans un processeur ayant des registres conçus pour exécuter des procédures avec la plus grande des tailles de mot selon la revendication 25, dans lequel les moyens pour placer l'indication conservée dans au moins un registre d'un certain nombre de registres du processeur désignant la taille de mot de la procédure dans la portion des zones de sauvegarde ayant la même adresse à chaque fois qu'une opération de sauvegarde intervient comprennent également des moyens pour placer une indication dans un bit autrement non-significatif de la portion conservant l'adresse de pointeur de pile.

31. Dispositif pour exécuter des procédures informatiques de tailles de mot différentes dans un processeur ayant des registres conçus pour exécuter des procédures avec la plus grande des tailles de mot selon la revendication 30, dans lequel les moyens pour placer l'indication dans un bit autrement non-significatif de la portion conservant l'adresse de pointeur de pile comprennent des moyens pour placer le bit dans une position binaire de poids faible de l'adresse de pointeur de pile.

32. Dispositif pour exécuter des procédures informatiques de tailles de mot différentes dans un processeur ayant des registres conçus pour exécuter des procédures avec la plus grande des tailles de mot selon la revendication 17, comprennent également des moyens pour restaurer l'information destinée aux registres à partir de la zone de sauvegarde désignée par l'indication.

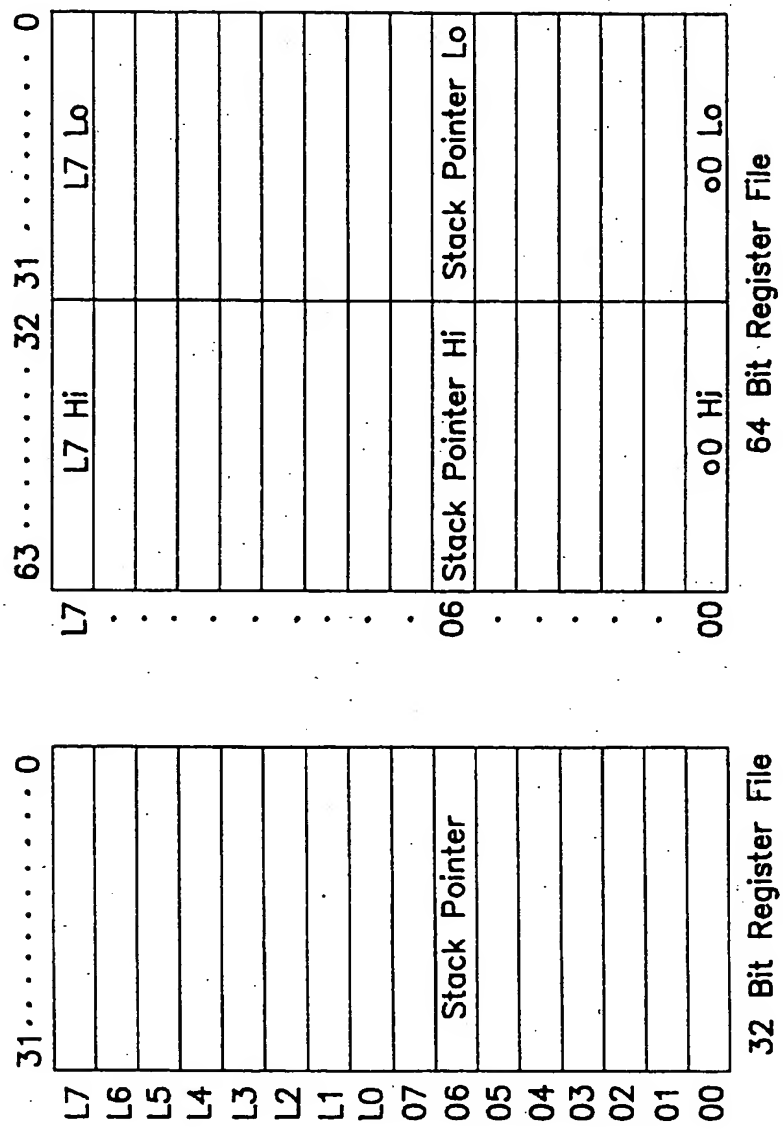
63 . . . . . 32 31 0
L7 Hi
L7 Lo
Stack Pointer Hi
Stack Pointer Lo
o0 Hi
o0 Lo
L7
06
00
64 Bit Register File
31 0
Stack Pointer
L7
L6
L5
L4
L3
L2
L1
L0
07
06
05
04
03
02
01
00
32 Bit Register File

FIG. 1

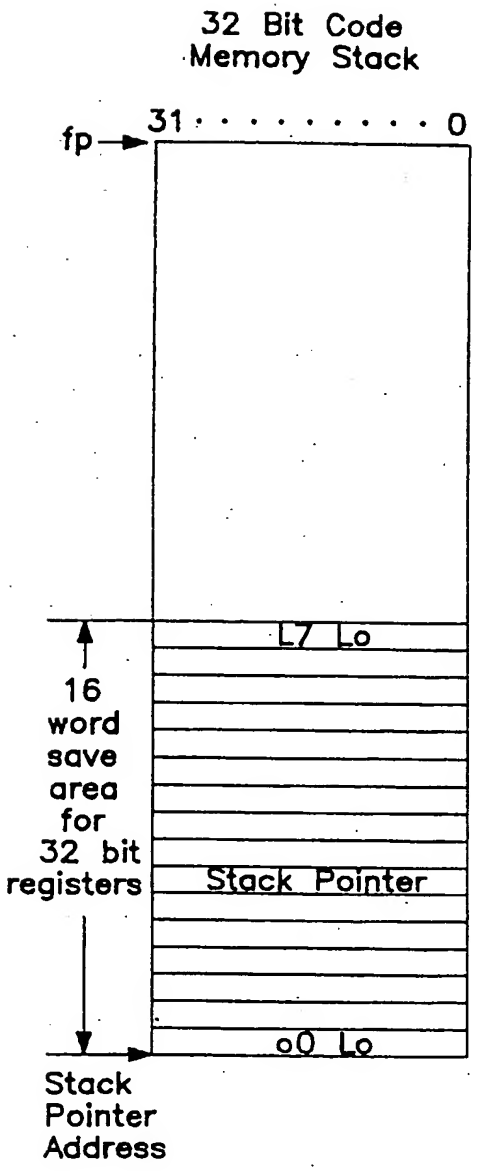
32 Bit Code
Memory Stack
fp
31 0
L7 Lo
16
word
save
area
for
32 bit
registers
Stack Pointer
o0 Lo
Stack
Pointer
Address

FIG. 2

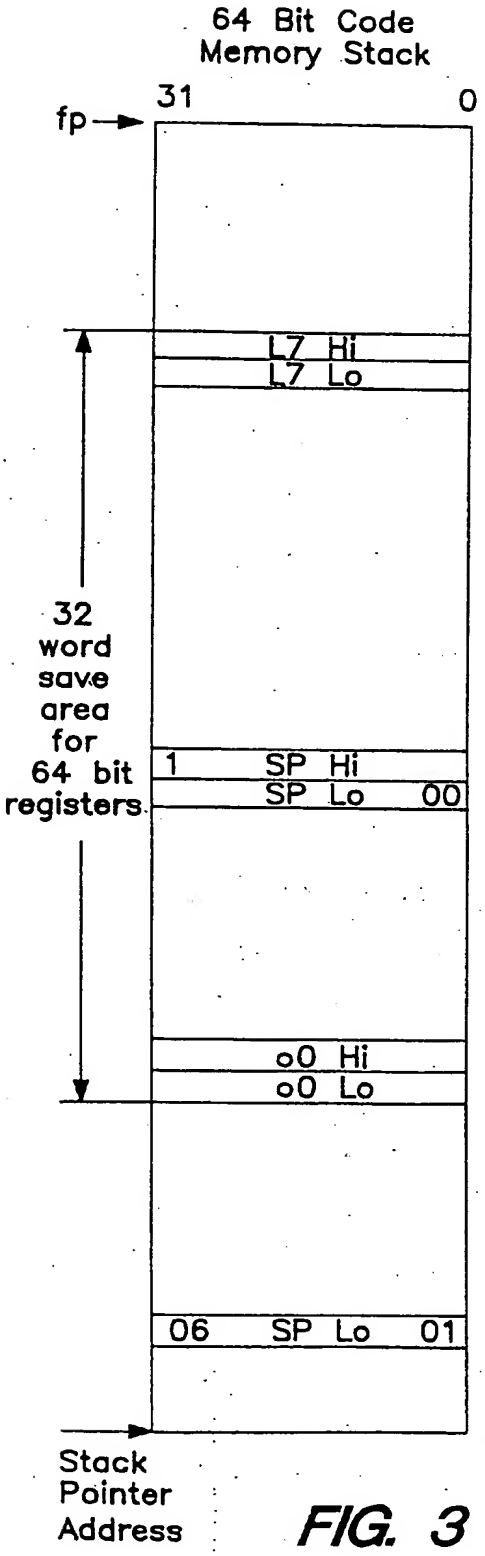
64 Bit Code
Memory Stack
fp
31
0
L7 Hi
L7 Lo
32
word
save
area
for
64 bit
registers
1 SP Hi
SP Lo 00
o0 Hi
o0 Lo
06 SP Lo 01
Stack
Pointer
Address

FIG. 3

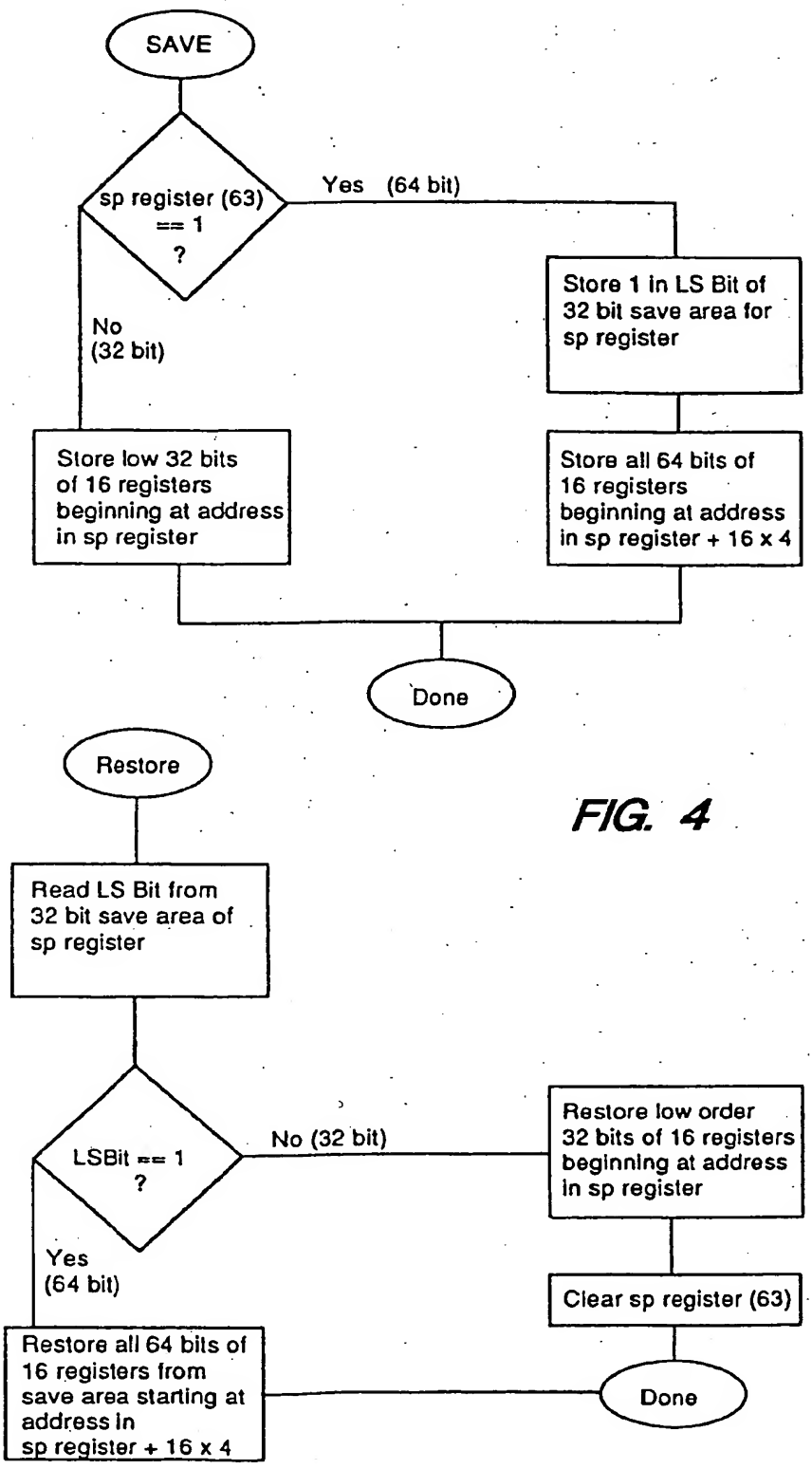
SAVE
sp register (63) == 1 ?
Yes (64 bit)
No (32 bit)
Store 1 in LS Bit of 32 bit save area for sp register
Store low 32 bits of 16 registers beginning at address in sp register
Store all 64 bits of 16 registers beginning at address in sp register + 16 x 4
Done
Restore
Read LS Bit from 32 bit save area of sp register
LSBit == 1 ?
No (32 bit)
Yes (64 bit)
Restore low order 32 bits of 16 registers beginning at address in sp register
Clear sp register (63)
Restore all 64 bits of 16 registers from save area starting at address in sp register + 16 x 4
Done

FIG. 4